Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Звіт

до лабораторної роботи №7

З дисципліни: «Кросплатформенні засоби програмування»

На тему: «Дослідження базових конструкцій мови Python»

Варіант 8

Виконав:

Киянець А.М.

Ст. групи КІ- 306

Прийняв:

Іванов Ю.С.

Львів 2023

Мета

Оволодіти навиками параметризованого програмування мовою Python.

Завдання

1. Написати та налагодити програму на мові Python згідно варіанту. Програма має задовольняти наступним вимогам:
   - програма має розміщуватися в окремому модулі;
   - програма має генерувати зубчатий список, який міститиме лише заштриховані області квадратної матриці згідно варіанту;
   - розмір квадратної матриці і символ-заповнювач масиву вводяться з клавіатури;
   - при не введені або введенні кількох символів-заповнювачів відбувається коректне переривання роботи програми;
   - сформований масив вивести на екран;
   - програма має містити коментарі.
2. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.
4. Дати відповідь на контрольні запитання.

Варіант

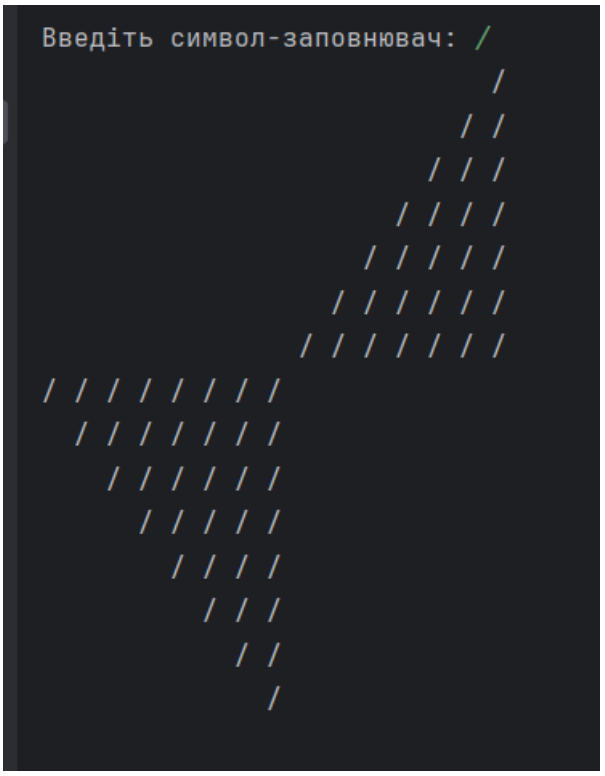
8	

Код програми

```
import struct  # Модуль для роботи з бінарними даними
import pickle


def generate_toothed_list(size, fill_char):
    """
    Генерує зубчатий список, який починається з правого верхнього краю
    і послідовно зростає до середини, а від середини зменшується в правий нижній
край.

    :param size: Розмір квадратної матриці.
    :param fill_char: Символ-заповнювач.
    :return: Зубчатий список.
    """
    toothed_list = []

    # Генеруємо зубчатий список
    for i in range(size):
        row = []  # Ініціалізуємо новий рядок
        for j in range(size):
            # Додаємо fill_char, якщо знаходимося в заштрихованій області, інакше
додаємо пробіл
            if j >= i and j < size - i:
                row.append(fill_char)
            else:
                row.append(" ")
        toothed_list.append(row)

    return toothed_list

def print_rotated_toothed_list(toothed_list):
```

```
    """
    Виводить обернутий на 90 градусів за годинниковою стрілкою зубчатий список.

    :param toothed_list: Зубчатий список.
    """
    for row in zip(*toothed_list[::-1]):
        print(" ".join(row))


def print_rotated_half_toothed_list(toothed_list):
    """
    Виводить обернутий на 90 градусів за годинниковою стрілкою та тільки першу
половину зубчатий список.

    :param toothed_list: Зубчатий список.
    """
    half_size = len(toothed_list[0]) // 2  # Розмір половини матриці

    rotated_rows = list(zip(*toothed_list[::-1]))

    for row in rotated_rows[:half_size]:
        print(" ".join(row))


def print_rotated_bottom_half_toothed_list(toothed_list):
    """
    Виводить обернутий на 90 градусів за годинниковою стрілкою та тільки нижню
половину зубчатий список,
    здійснюючи зсув вліво.

    :param toothed_list: Зубчатий список.
    """
    half_size = len(toothed_list[0]) // 2  # Розмір половини матриці

    rotated_rows = list(zip(*toothed_list[::-1]))

    for row in rotated_rows[half_size:]:
        # Забираємо пробіли в ліво для нижньої половини рядків
        shifted_row = row[half_size:]  # Fix here
        print(" ".join(shifted_row))

def main():
    try:
        # Зчитування числа з файлу output.bin
        with open(r'D:\LPNU\Chapter_5\KZP\KZP\Lab_8\output.bin', 'rb') as file:
            size = pickle.load(file)
            # Використовуємо struct.unpack для розпакування бінарних даних
        size = int(abs(size))
        print(size)
        #size = int(input("Введіть розмір квадратної матриці: "))
        fill_char = input("Введіть символ-заповнювач: ")

        # Генеруємо зубчатий список
        toothed_list = generate_toothed_list(size, fill_char)

        # Виводимо обернутий на 90 градусів за годинниковою стрілкою зубчатий
список на екран
        #print_rotated_toothed_list(toothed_list)
        print_rotated_half_toothed_list(toothed_list)
        print_rotated_bottom_half_toothed_list(toothed_list)

    except ValueError:
        print("Помилка: Неправильний формат введених даних.")
    except KeyboardInterrupt:
        print("\nРоботу програми перервано користувачем.")

# Викликаємо головну функцію, якщо файл виконується самостійно
if __name__ == "__main__":
    main()
```

Результат виконання роботи після захисту



Висновок

Я ознайомився з основними принципами мови програмування python та оволодів навиками застосування них. Закріпив теорію на практиці.